

FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA

```
MM      MM      AAAAAA      KK      KK      NN      NN      MM      MM      BBBBBBBBBB
MM      MM      AAAAAA      KK      KK      NN      NN      MM      MM      BBBBBBBBBB
MMMM    MMMM    AA      AA      KK      KK      NN      NN      MMMM    MMMM    BB      BB
MMMM    MMMM    AA      AA      KK      KK      NN      NN      MMMM    MMMM    BB      BB
MM      MM      AA      AA      KK      KK      NNNN      NN      MM      MM      BB      BB
MM      MM      AA      AA      KK      KK      NNNN      NN      MM      MM      BB      BB
MM      MM      AA      AA      KKKKKK      NN      NN      NN      MM      MM      BBBBBBBBBB
MM      MM      AA      AA      KKKKKK      NN      NN      NN      MM      MM      BBBBBBBBBB
MM      MM      AAAAAA      KK      KK      NN      NN      NN      MM      MM      BB      BB
MM      MM      AAAAAA      KK      KK      NN      NN      NN      MM      MM      BB      BB
MM      MM      AA      AA      KK      KK      NN      NN      MM      MM      BB      BB
MM      MM      AA      AA      KK      KK      NN      NN      MM      MM      BB      BB
MM      MM      AA      AA      KK      KK      NN      NN      MM      MM      BBBBBBBBBB
MM      MM      AA      AA      KK      KK      NN      NN      MM      MM      BBBBBBBBBB
```

```
....
....
....
....
```

```
LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLLLL      IIIIII      SSSSSSSS
```


(2)	65	Local constants and flags
(3)	96	MAKE_NAMEBLOCK - Build RAD-50 name block
(4)	333	TYPE - Determine character class


```
0000 1 .TITLE MAKNMB - Build the name block
0000 2 .IDENT /V04-000/
0000 3
0000 4 *****
0000 5 :
0000 6 :* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 :* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 :* ALL RIGHTS RESERVED.
0000 9 :
0000 10 :* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 :* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 :* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 :* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 :* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 :* TRANSFERRED.
0000 16 :
0000 17 :* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 :* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 :* CORPORATION.
0000 20 :
0000 21 :* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 :* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23 :
0000 24 :
0000 25 :*****
0000 26 :
0000 27 :++
0000 28 :
0000 29 : FACILITY: F11ACP Structure Level 1
0000 30 :
0000 31 : ABSTRACT:
0000 32 :
0000 33 : This routine converts the specified file name string into the RAD-50
0000 34 : name block format. It also saves the original pattern in case
0000 35 : character wildcarding is used.
0000 36 :
0000 37 : ENVIRONMENT:
0000 38 :
0000 39 : VAX/VMS operating system, including privileged system services
0000 40 : and internal exec routines.
0000 41 :
0000 42 :--
0000 43 :
0000 44 : AUTHOR: L. Mark Pilant, CREATION DATE: 23-Apr-1984 10:00
0000 45 : (Original BLISS version by Andy Goldstein)
0000 46 :
0000 47 : MODIFIED BY:
0000 48 :
0000 49 : V03-001 LMP0292 L. Mark Pilant, 2-Aug-1984 11:54
0000 50 : Fix a bug that caused digits in the file type field to be
0000 51 : garbled.
0000 52 :
0000 53 :**
0000 54 :
0000 55 :
0000 56 : INCLUDE FILES:
0000 57 :
```


MAKNMB
V04-000

- Build the name block

D 2

16-SEP-1984 00:43:57 VAX/VMS Macro V04-00
6-SEP-1984 09:13:54 [F11A.SRC]MAKNMB.MAR;1

Page 2
(1)

```
0000 58 ;      FCPPRE.MAR
0000 59
0000 60 ; Structure and offset definitions.
0000 61
0000 62      $FIBDEF
0000 63      $NMBDEF
```



```
0000 65 .SBTTL Local constants and flags
0000 66
0000 67 ; Character type codes. These are used to determine the action to take.
0000 68 ; (These constants are determined by the offsets into the type tables
0000 69 ; below.)
0000 70
00000001 0000 71 PERCENT= 1 ; Percent sign (%)
00000002 0000 72 STAR= 2 ; Asterisk (*)
00000003 0000 73 DIGIT= 3 ; Numeric digit
00000004 0000 74 LC_ALPHA= 4 ; Lower case alphabetic
00000005 0000 75 UC_ALPHA= 5 ; Upper case alphabetic
00000006 0000 76 DOT= 6 ; Period (.)
00000007 0000 77 SEMI= 7 ; Semi-colon (;)
0000 78
0000 79 ; Tables used to determine the above offsets:
0000 80
00000000 0000 81 .PSECT $CODE$,NOWRT,2
0000 82
0000 83 LO_CHR_TABLE:
3B 2E 41 61 30 2A 25 00 0000 84 .ASCII <0>/%*0aA.;/
0008 85 HI_CHR_TABLE:
3B 2E 5A 7A 39 2A 25 00 0008 86 .ASCII <0>/%*9zZ.;/
0010 87
0010 88 ;
0010 89 ;
0010 90 ;
0010 91 ;
0010 92 ;
0010 93 ;
0010 94 ;

Special note:

Throughout the following routine, the following register
assignments are used:

R6 = the number of characters remaining in the string
R7 = the address of the next character
```



```
0010 96 .SBTTL MAKE_NAMEBLOCK - Build RAD-50 name block
0010 97
0010 98 :++
0010 99
0010 100 : FUNCTIONAL DESCRIPTION:
0010 101
0010 102 : This routine converts a file name string into the RAD-50 name block
0010 103 : format.
0010 104
0010 105 : CALLING SEQUENCE:
0010 106 : MAKE_NAMEBLOCK (ARG1, ARG2, ARG3, ARG4)
0010 107
0010 108 : INPUT PARAMETERS:
0010 109 : ARG1: address of FIB if pattern parse
0010 110 : 0 if resultant file string parse
0010 111 : ARG2: length of file name string
0010 112 : ARG3: address of file name string
0010 113
0010 114 : IMPLICIT INPUTS:
0010 115 : NONE
0010 116
0010 117 : OUTPUT PARAMETERS:
0010 118 : ARG4: address of file name block
0010 119
0010 120 : IMPLICIT OUTPUTS:
0010 121 : NONE
0010 122
0010 123 : ROUTINE VALUE:
0010 124 : NONE
0010 125
0010 126 : SIDE EFFECTS:
0010 127 : NONE
0010 128
0010 129 :--
0010 130
00000004 0010 131 FIB= 4 ; ARG list offsets
00000008 0010 132 LENGTH= 8
0000000C 0010 133 STRING= 12
00000010 0010 134 NAME_BLOCK= 16
0010 135
00FFC 0010 136 .ENTRY MAKE_NAMEBLOCK, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0012 137
0012 138 MOVL NAME_BLOCK(AP),R9 ; Get address of name block
0016 139 MOVCS #0,(SP),#0,#NMB$C_LENGTH,(R9) ; Clear out block
001C 140 MOVL STRING(AP),R7 ; Get address of first character
0020 141 LOCC #^A/ /,LENGTH(AP),(R7) ; Find the end of the string
0025 142 SUBL3 R7,R1,R6 ; Calculate size of the string
0029 143
0029 144 MOVAB NMB$W_NAME(R9),R10 ; Where RAD-50 name goes
002D 145 MOVAB NMB$T_ASCNAMTXI(R9),R11 ; Where ASCII pattern goes
0031 146 PUSHL R11 ; Save current pattern pointer
0033 147
0033 148 ; Build the file name field. This consists of 3 words of 3 RAD-50 characters
0033 149 ; in each word.
0033 150
0033 151 MOVL #3,R3 ; Number of words to do
0036 152 CLRL R4 ; Clear wildcard flag
```

69 28 00 59 10 AC D0 0012 138
57 6E 00 2C 0016 139
67 08 AC 20 3A 001C 140
56 51 57 C3 0020 141
5A 06 A9 9E 0029 142
5B 13 A9 9E 002D 143
5B DD 0031 144
0033 145
0033 146
0033 147
53 03 D0 0033 148
54 D4 0036 149


```
- Build the name block
MAKE_NAMEBLOCK - Build RAD-50 name block

55 03 D0 0038 153 10$: MOVL #3,R5 ; Number of characters to pack
6A 28 A4 003B 154 20$: MULW2 #40,(R10) ; Make room for RAD-50 char
51 67 9A 003E 155 MOVZBL (R7),R1 ; Get next source string char
07 00 016A 30 0041 156 BSBW TYPE ; Determine character class
50 50 CF 0044 157 30$: CASEL R0,#0,#7 ; Dispatch on character class
0030' 0048 158 40$: .WORD 110$-40$ ; End of the string
0010' 004A 159 .WORD 50$-40$ ; Percent sign
0010' 004C 160 .WORD 50$-40$ ; Asterisk
0015' 004E 161 .WORD 60$-40$ ; Digit
001B' 0050 162 .WORD 70$-40$ ; Lower case alphabetic
001E' 0052 163 .WORD 80$-40$ ; Upper case alphabetic
0030' 0054 164 .WORD 110$-40$ ; Period
0030' 0056 165 .WORD 110$-40$ ; Semi-colon
54 50 C0 0058 166 50$: ADDL2 R0,R4 ; Else note the wildcard character
12 11 005B 167 BRB 100$ ; Go get another character
50 51 12 A3 005D 168 60$: SUBW3 #^A/0/-30,R1,R0 ; Convert digit to RAD-50
09 11 0061 169 BRB 90$ ; Go save RAD-50 character
51 20 A2 0063 170 70$: SUBW2 #^X20,R1 ; Convert lower to upper case
50 8F A3 0066 171 80$: SUBW3 #^A/A/-1,R1,R0 ; Convert alphabetic to RAD-50
6A 50 A0 006C 172 90$: ADDW2 R0,(R10) ; Accumulate characters
8B 51 90 006F 173 100$: MOVB R1,(R11)+ ; Save ASCII character
56 D7 0072 174 DECL R6 ; One less character
02 15 0074 175 BLEQ 110$ ; Xfer if no more
57 D6 0076 176 INCL R7 ; Else update pointer also
C0 55 F5 0078 177 110$: SOBGTR R5,20$ ; Continue till word full
8A B5 007B 178 TSTW (R10)+ ; Advance to next RAD-50 word
BB 53 F5 007D 179 SOBGTR R3,10$ ; Continue till all full
0080 180
0080 181 ; Check for FIB$V ALLNAM set and the first character of the file name string
0080 182 ; being a dot. This hack is used by the compatibility mode program PIP.
0080 183
58 04 AC D0 0080 184 MOVL FIB(AP),R8 ; Get the address of the FIB
08 14 A8 0D 13 0084 185 BEQL 120$ ; Skip following if no FIB given
5B 05 E1 0086 186 BBC #FIB$V_ALLNAM,FIB$W_NMCTL(R8),120$ ; Xfer if file name present
8B 6E D0 008B 187 MOVL (SP),R11 ; Get saved pattern pointer
2A 90 008E 188 MOVB #^A/*/, (R11)+ ; Save a wild name character
12 11 0091 189 BRB 125$ ; Go note wildcard use
0093 190
0093 191 ; Set any applicable wildcard flags.
0093 192
50 8E 01 C1 0093 193 120$: ADDL3 #1,(SP)+,R0 ; Clean stack & set up for below
54 D5 0097 194 TSTL R4 ; Any wildcards at all?
14 13 0099 195 BEQL 140$ ; Xfer if no wildcards at all
02 54 D1 009B 196 CMPL R4,#STAR ; Single asterisk?
09 12 009E 197 BNEQ 130$ ; Xfer if not
5B 50 D1 00A0 198 CMPL R0,R11 ; See if asterisk is alone
04 12 00A3 199 BNEQ 130$ ; Xfer if not alone
10 A9 20 A8 00A5 200 125$: BISW2 #NMBSM_ALLNAM,NMBSW_FLAGS(R9) ; Else note wild field
10 A9 0100 8F A8 00A9 201 130$: BISW2 #NMBSM_WILD,NMBSW_FLAGS(R9) ; Note presence of wildcards
00AF 202
00AF 203 ; Now that the file name field has been built, check the delimiting character.
00AF 204 ; The only legal characters are period and semi-colon.
00AF 205
51 67 9A 00AF 206 140$: MOVZBL (R7),R1 ; Get the delimiting character
07 00 00F9 30 00B2 207 BSBW TYPE ; Get character class
50 50 CF 00B5 208 CASEL R0,#0,#7 ; Dispatch on character class
0019' 00B9 209 150$: .WORD 170$-150$ ; End of the string
```



```
00ED' 00BB 210 .WORD BADFILENAME-150$ ; Percent sign
00ED' 00BD 211 .WORD BADFILENAME-150$ ; Asterisk
00ED' 00BF 212 .WORD BADFILENAME-150$ ; Digit
00ED' 00C1 213 .WORD BADFILENAME-150$ ; Lower case alphabetic
00ED' 00C3 214 .WORD BADFILENAME-150$ ; Upper case alphabetic
0010' 00C5 215 .WORD 160$-150$ ; Period
0019' 00C7 216 .WORD 170$-150$ ; Semi-colon
8B 51 90 00C9 217 160$: MOVB R1,(R11)+ ; Else save ASCII character
56 D7 00CC 218 DECL R6 ; One less character
57 D6 00CE 219 INCL R7 ; Next position in the buffer
03 11 00D0 220 BRB 180$ ; Go check for PIP hack
00D2 221
00D2 222 ; Make sure that the file name is properly delimited.
00D2 223
8B 2E 90 00D2 224 170$: MOVB #^A/./,(R11)+ ; Else terminate file name properly
00D5 225
00D5 226 ; Build the file type field. This consists of 3 RAD-50 characters.
00D5 227
5B DD 00D5 228 180$: PUSHL R11 ; Save current pattern pointer
54 D4 00D7 229 CLRL R4 ; Clear wildcard flag
55 03 D0 00D9 230 MOVL #3,R5 ; Number of characters to pack
6A 28 A4 00DC 231 190$: MULW2 #40,(R10) ; Make room for RAD-50 char
51 67 9A 00DF 232 MOVZBL (R7),R1 ; Get next source string char
07 00 00C9 30 00E2 233 BSBW TYPE ; Determine character class
50 50 CF 00E5 234 200$: CASEL R0,#0,#7 ; Dispatch on character class
0030' 00E9 235 210$: .WORD 280$-210$ ; End of the string
0010' 00EB 236 .WORD 220$-210$ ; Percent sign
0010' 00ED 237 .WORD 220$-210$ ; Asterisk
0015' 00EF 238 .WORD 230$-210$ ; Digit
001B' 00F1 239 .WORD 240$-210$ ; Lower case alphabetic
001E' 00F3 240 .WORD 250$-210$ ; Upper case alphabetic
0030' 00F5 241 .WORD 280$-210$ ; Period
0030' 00F7 242 .WORD 280$-210$ ; Semi-colon
54 50 C0 00F9 243 220$: ADDL2 R0,R4 ; Else note the wildcard character
12 11 00FC 244 BRB 270$ ; Go get another character
50 51 12 A3 00FE 245 230$: SUBW3 #^A/0/-30,R1,R0 ; Convert digit to RAD-50
09 11 0102 246 BRB 260$ ; Go save RAD-50 character
51 20 A2 0104 247 240$: SUBW2 #^X20,R1 ; Convert lower to upper case
50 51 0040 8F A3 0107 248 250$: SUBW3 #^A/A/-1,R1,R0 ; Convert alphabetic to RAD-50
6A 50 A0 010D 249 260$: ADDW2 R0,(R10) ; Accumulate characters
8B 51 90 0110 250 270$: MOVB R1,(R11)+ ; Save ASCII character
56 D7 0113 251 DECL R6 ; One less character
02 15 0115 252 BLEQ 280$ ; Xfer if no more
57 D6 0117 253 INCL R7 ; Else update pointer also
C0 55 F5 0119 254 280$: SOBGTR R5,190$ ; Continue till word full
011C 255
011C 256 ; Check for FIB$V_ALLTYP set and the first character of the file type
011C 257 ; string being a dot or a semi-colon. This hack is used by the compatability
011C 258 ; mode program PIP.
011C 259
58 04 AC D0 011C 260 MOVL FIB(AP),R8 ; Get the address of the FIB
0D 13 0120 261 BEQL 290$ ; Skip following if no FIB given
08 14 A8 04 E1 0122 262 BBC #FIB$V_ALLTYP,FIB$W_NMCTL(R8),290$ ; Xfer if file name present
5B 6E D0 0127 263 MOVL (SP),RT1 ; Get saved pattern pointer
8B 2A 90 012A 264 MOVB #^A/*/,(R11)+ ; Else save a wild type character
12 11 012D 265 BRB 300$ ; Go note wildcard use
012F 266
```



```
012F 267 ; Set any applicable wildcard flags.
012F 268
50 8E 01 C1 012F 269 290$: ADDL3 #1,(SP)+,R0 ; Clean stack & set up for below
54 D5 0133 270 TSTL R4 ; Any wildcards at all?
14 13 0135 271 BEQL 320$ ; Xfer if no wildcards at all
02 54 D1 0137 272 CMPL R4,#STAR ; Single asterisk?
09 12 013A 273 BNEQ 310$ ; Xfer if not
5B 50 D1 013C 274 CMPL R0,R11 ; See if asterisk is alone
04 12 013F 275 BNEQ 310$ ; Xfer if not alone
10 A9 10 A8 0141 276 300$: BLSW2 #NMB$M_ALLTYP,NMB$W_FLAGS(R9) ; Else note wild field
10 A9 0100 8F A8 0145 277 310$: BLSW2 #NMB$M_WILD,NMB$W_FLAGS(R9) ; Note presence of wildcards
014B 278
014B 279 ; Now set the size of the pattern string.
014B 280
12 A9 50 13 A9 9E 014B 281 320$: MOVAB NMB$T_ASCNAMTXT(R9),R0 ; Get base address
5B 50 83 014F 282 SUBB3 R0,R11,NMB$B_ASCNAMSIZ(R9) ; Save size of pattern string
0154 283
0154 284 ; Now that the file name field has been built, check the delimiting character.
0154 285 ; The only legal characters are period, semi-colon, and end of string.
0154 286
51 67 9A 0154 287 MOVZBL (R7),R1 ; Get the delimiting character
0054 30 0157 288 BSBW TYPE ; Get character class
07 00 50 CF 015A 289 CASEL R0,#0,#7 ; Dispatch on character class
0047' 015E 290 330$: .WORD 380$-330$ ; End of the string
0048' 0160 291 .WORD BADFILENAME-330$ ; Percent sign
0048' 0162 292 .WORD BADFILENAME-330$ ; Asterisk
0048' 0164 293 .WORD BADFILENAME-330$ ; Digit
0048' 0166 294 .WORD BADFILENAME-330$ ; Lower case alphabetic
0048' 0168 295 .WORD BADFILENAME-330$ ; Upper case alphabetic
0010' 016A 296 .WORD 340$-330$ ; Period
0010' 016C 297 .WORD 340$-330$ ; Semi-colon
56 D7 016E 298 340$: DECL R6 ; One less character
33 15 0170 299 BLEQ 380$ ; Xfer if nothing left to parse
57 D6 0172 300 INCL R7 ; Next position in the buffer
0174 301
0174 302 ; Now for the version. If the version is not wild, and there are characters
0174 303 ; left to parse, try to get the binary version number. If this fails, note
0174 304 ; the error.
0174 305
01 56 D1 0174 306 360$: CMPL R6,#1 ; Check for only one char
0D 14 0177 307 BGTR 370$ ; Xfer if more than one char
2A 67 91 0179 308 CMPB (R7),#^A/*/ ; Else check for a wildcard
08 12 017C 309 BNEQ 370$ ; Xfer if not wild
10 A9 0108 8F A8 017E 310 BLSW2 #NMB$M_WILD!NMB$M_ALLVER,NMB$W_FLAGS(R9) ; Else note wild
1F 11 0184 311 BRB 380$ ; Go finish up
7E 5E D4 0186 312 370$: CLRL -(SP) ; Storage for converted version
7E 56 7D 0188 313 MOVL SP,-(SP) ; Save address of storage
00000000'GF 03 FB 018E 315 CALLS #3,G^LIB$CVT DTB ; Save count and address
OE 50 E9 0195 316 BLBC R0,BADFILENAME ; Convert to binary
00007FFF 8F 6E D1 0198 317 CMPL (SP),#32767 ; Xfer if any errors
09 1A 019F 318 BGTRU BADFILEVER ; Else range check
OE A9 8E F7 01A1 319 CVTLW (SP)+,NMB$W_VERSION(R9) ; Xfer if bad version number
01A5 320 ; Else save version number
01A5 321 ; All is done. Return to the caller.
01A5 322
04 01A5 323 380$: RET
```



```
01A6 324  
01A6 325 ; Error returns.  
01A6 326  
01A6 327 BADFILENAME:  
01A6 328 ERR_EXIT #SS$_BADFILENAME  
01AA 329  
01AA 330 BADFILEVER:  
01AA 331 ERR_EXIT #SS$_BADFILEVER
```



```
01AE 333      .SBTTL TYPE - Determine character class
01AE 334
01AE 335      :++
01AE 336      :
01AE 337      : FUNCTIONAL DESCRIPTION:
01AE 338      :
01AE 339      :       This routine determines the type code of the current character
01AE 340      :       in the string.
01AE 341      :
01AE 342      : CALLING SEQUENCE:
01AE 343      :       TYPE ( )
01AE 344      :
01AE 345      : INPUT PARAMETERS:
01AE 346      :       NONE
01AE 347      :
01AE 348      : IMPLICIT INPUTS:
01AE 349      :       R6: number of characters left in string
01AE 350      :       R7: string pointer
01AE 351      :
01AE 352      : OUTPUT PARAMETERS:
01AE 353      :       NONE
01AE 354      :
01AE 355      : IMPLICIT OUTPUTS:
01AE 356      :       NONE
01AE 357      :
01AE 358      : ROUTINE VALUE:
01AE 359      :       type code of character:
01AE 360      :       0: end of string
01AE 361      :       1: percent
01AE 362      :       2: star
01AE 363      :       3: numeric
01AE 364      :       4: lower case alpha
01AE 365      :       5: upper case alpha
01AE 366      :       6: dot
01AE 367      :       7: semicolon
01AE 368      :
01AE 369      : SIDE EFFECTS:
01AE 370      :       NONE
01AE 371      :
01AE 372      :--
01AE 373
01AE 374 TYPE:   CLRL      R0      ; Assume nothing left
01AE 375      TSTL      R6      ; Correct assumption?
01AE 376      BLEQ      30$     ; Xfer if so
01AE 377      MOVL      #1,R0   ; Set initial index
01AE 378 10$:    CMPB      (R7),LO_CHR_TABLE[R0] ; Within limits?
01AE 379      BLSSU      20$     ; Xfer if not
01AE 380      CMPB      (R7),HI_CHR_TABLE[R0]
01AE 381      BLEQU      30$
01AE 382 20$:    AOBLEQ     #7,R0,10$ ; Continue till end
01AE 383      BRW      BADFILENAME ; Illegal if off the end
01AE 384 30$:    RSB
01AE 385
01AE 386      .END
```

50	D4	01AE	374	TYPE:	CLRL	R0	; Assume nothing left
56	D5	01B0	375		TSTL	R6	; Correct assumption?
1A	15	01B2	376		BLEQ	30\$	; Xfer if so
50	01	01B4	377		MOVL	#1,R0	; Set initial index
FE43 CF40	67	01B7	378	10\$:	CMPB	(R7),LO_CHR_TABLE[R0]	; Within limits?
	08	01BD	379		BLSSU	20\$	; Xfer if not
FE43 CF40	67	01BF	380		CMPB	(R7),HI_CHR_TABLE[R0]	
	07	01C5	381		BLEQU	30\$	
EC 50	07	01C7	382	20\$:	AOBLEQ	#7,R0,10\$	; Continue till end
	FFD8	01CB	383		BRW	BADFILENAME	; Illegal if off the end
	05	01CE	384	30\$:	RSB		; Return with class in R0
		01CF	385				
		01CF	386		.END		

MAKNMB
Symbol table

- Build the name block

L 2

16-SEP-1984 00:43:57 VAX/VMS Macro V04-00
6-SEP-1984 09:13:54 [F11A.SRC]MAKNMB.MAR;1

Page 10
(4)

AQB_TYPE	= 00000005		
BADFILENAME	000001A6	R	02
BADFILEVER	000001AA	R	02
BITMAP_TYPE	= 00000001		
DIGIT	= 00000003		
DIRECTORY_TYPE	= 00000002		
DOT	= 00000006		
FCB_TYPE	= 00000000		
FIB	= 00000004		
FIBSV_ALLNAM	= 00000005		
FIBSV_ALLTYP	= 00000004		
FIBSW_NMCTL	= 00000014		
HEADER_TYPE	= 00000000		
HI CHR_TABLE	00000008	R	02
INDEX_TYPE	= 00000003		
LC_ALPHA	= 00000004		
LENGTH	= 00000008		
LIBSCVT DTB	*****	X	02
LO CHR_TABLE	00000000	R	02
MAKE_NAMEBLOCK	00000010	RG	02
MVL_TYPE	= 00000004		
NAME_BLOCK	= 00000010		
NMB\$B_ASCNAMSIZ	= 00000012		
NMB\$C_LENGTH	= 00000028		
NMB\$M_ALLNAM	= 00000020		
NMB\$M_ALLTYP	= 00000010		
NMB\$M_ALLVER	= 00000008		
NMB\$M_WILD	= 00000100		
NMB\$T_ASCNAMTXT	= 00000013		
NMB\$W_FLAGS	= 00000010		
NMB\$W_NAME	= 00000006		
NMB\$W_VERSION	= 0000000E		
PERCENT	= 00000001		
RVT_TYPE	= 00000003		
SEMI	= 00000007		
SS\$_BADFILENAME	*****	X	02
SS\$_BADFILEVER	*****	X	02
STAR	= 00000002		
STRING	= 0000000C		
TYPE	000001AE	R	02
UC_ALPHA	= 00000005		
VCB_TYPE	= 00000002		
WCB_TYPE	= 00000001		

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$CODE\$	000001CF (463.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.12	00:00:00.45
Command processing	124	00:00:00.73	00:00:04.26
Pass 1	190	00:00:03.36	00:00:09.97
Symbol table sort	0	00:00:00.38	00:00:00.60
Pass 2	82	00:00:01.22	00:00:04.04
Symbol table output	7	00:00:00.05	00:00:00.05
Psect synopsis output	1	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	435	00:00:05.88	00:00:19.47

The working set limit was 1050 pages.

18307 bytes (36 pages) of virtual memory were used to buffer the intermediate code.

There were 20 pages of symbol table space allocated to hold 245 non-local and 41 local symbols.

489 source lines were read in Pass 1, producing 17 object records in Pass 2.

13 pages of virtual memory were used to define 12 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	1
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	5

278 GETS were required to define 5 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:MAKNMB/OBJ=OBJ\$:MAKNMB MSRC\$:FCPPRE/UPDATE=(ENH\$:FCPPRE)+MSRC\$:MAKNMB/UPDATE=(ENH\$:MAKNMB)+EXECML\$/LIB

0166 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

MODIFY
LIS

REQUEU
LIS

RWATTR
LIS

SCHFCB
LIS

MAKACC
LIS

MPWIND
LIS

MAPUBN
LIS

PMS
LIS

RDHEDR
LIS

RWUB
LIS

RETDIR
LIS

ROBLOK
LIS

SMALOC
LIS

MOUNT
LIS

MAKMBE
LIS

MAKSTR
LIS

MAKTHOR
LIS